IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

Linear Threshold Tunneling Technology based Logic Gates, Full Adder and a Converter

Anup Kumar Biswas,
Department of Computer Science and Engineering
Kalyani Govt. Engineering College, Kalyani,
Nadia-741235, West Bengal, India

ABSTRACT

In business, engineering, science and technology, electronic goods of Low cost, low power consumption, high operating speed, and high integration density are economically requisite in the present era. Single Electron tunneling based threshold gate is one approach with which we are able to implement all logic gates, combinational and sequential circuits. Threshold Logic Gates (TLGs) and Single Electron tunneling devices (SEDs) bear the capabilities of controlling the transport of an electron through a tunnel junction at a particular time. A single electron containing the charge is sufficient to store an information in a SED. Power that is required in the single electron tunneling circuits is really low in comparison with the (CMOS) circuits. The processing delay is very low and speed of the processing of TLG based devices will be close to electronic speed. The single-electron transistor (SET) and TLG are attracting scientists, technologists and researchers to design and implement for the consuming of ultra-low power and their small sizes. All tunneling events in a TLG-based circuit happens when only a single electron tunnels from one conductor to other through the tunnel junction when the bias voltage and multiple input signal voltages are applied. For implementing logic gates, Full adder and a converter, TLG would be a best candidate to fulfill the necessities needed for their implementations. So far as an Ultra-low noise is concerned, TLG based circuit can be considered to be a best selection for implementing the desired tunneling circuits. Different TLGs like 2-input AND, 2-input OR, 3-input AND/NAND, 3-input OR/NOR, XOR/XNOR, Full adder, and a converter of BCD-to-Excess-3 have been implemented by using linear threshold logic gates or devices. Truth table or simulated results of them are given in parallel in due places.

Key words: up-down Counter, Electron-tunneling, Coulomb-blockade, linear threshold gate

1. INTRODUCTION

From the point of view of semiconductor technology that the ever decreasing feature size and the corresponding increase in density of transistors facilitates many improvements in semiconductor based designs. But one day such improvement will eventually come to an end. For ensuring further feature size reduction, possible imminent technologies with greater scaling potential like single electron tunneling technology is currently under the investigation of the researchers. SET-based circuits or threshold logic gate based circuits are constructed with tunnel junctions, through which an electron can be passed in a controlled manner.

Single Electron tunneling based device is one of such an equipment by which all Boolean logic gates and more complex circuits can be implemented. Tunneling events happen when a single electron can pass through the tunnel junction under the action of bias voltage and multiple input voltages connected to the islands via small capacitors. For implementing a BCD-to-Excess-3 counter, TLG would be a suitable candidate.

2. Multiple input threshold logic gate

A multiple input threshold logic gate [1, 2, 3, 4, 8, 9, 10] is a gate which is made up of a tunnel junction bearing a capacitance C_j and resistance R_j, two multiple input-signals V_k^Ps and V_l^ns connected at two points 'p' and 'q'. Each input voltage V_k^P, for the top left side, is connected to the point "q" through the corresponding capacitors $C_k^P s$ and each input voltage V_l^n, for the bottom left side, is connected to the point "p" through their respective capacitors C_l^ns. Supply voltage or Bias voltage V_b is connected to point the "b" through a capacitor C_b as well. Junction capacitor C_j is connected to point "p" that is grounded through another capacitor C_0. We will be able to implement the LTGs with the help of a function presented by the signun function(x) of h(x) expressed by equations (1) and (2).

$$g(x) = \text{sgn}\{h(x)\} = \begin{cases} 0, & if\ h(x) < 0 \\ 1, & if\ h(x) \geq 0 \end{cases} \quad \cdots\cdots (1)$$

$$h(x) = \sum_{k=1}^{n}(w_k \times x_k) - \theta \quad \cdots\cdots (2)$$

where x_k being the n-Boolean inputs and w_k being their corresponding n integer weights.

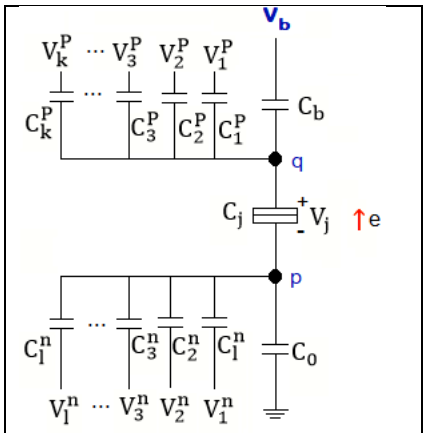


Fig. 1 Multiple input TLG

The LTG compares the weighted sum of the inputs $\sum_{k=1}^{n}(w_k \times x_k)$ with the threshold value θ, if the weighted sum-value becomes greater than or equal to the threshold or critical voltage value θ then the logic output of the LTG would be high (logical "1"), otherwise it would be low (logical "0").

The tunnel junction capacitance C_j and the capacitance C_0 are in series being considered the two basic circuit elements in a LTG. The input signal voltages $V_1^P, V_2^P, V_3^P, \ldots, V_k^P$, which are weighted by their corresponding vector capacitances $C_1^P, C_2^P, C_3^P, \ldots, C_k^P$, are added to the junction voltage, V_j. Whereas, the input signal voltages $V_1^n, V_2^n, V_3^n, \ldots, V_l^n$ (which are weighted by their corresponding vector capacitances $C_1^n, C_2^n, C_3^n, \ldots, C_l^n$), are being subtracted from the voltage, V_j.

The critical voltage V_c is needed to enable tunneling action, and which acts as the intrinsic threshold of the tunnel junction circuit. The supply or bias voltage V_b connected to tunnel junction through the capacitance, C_b, is used to adjust the gate threshold to the desired value θ. When a tunneling happens though the tunnel junction, an electron goes through the junction from p to q as directed by an arrow in Fig. 1.
The following notations would be used for the rest our discussion.

$$C_\Sigma^P = C_b + \sum_{k=1}^{g} C_k^P \quad \cdots\cdots (3)$$

$$C_\Sigma^n = C_0 + \sum_{l=1}^{h} C_l^n \quad \cdots\cdots (4)$$

$$C_T = C_\Sigma^P C_j + C_\Sigma^P C_\Sigma^n + C_j C_\Sigma^n \quad \cdots\cdots (5)$$

When we assume that all voltage sources in Fig. 1 are connected to ground, then the circuit can be treated as it is made up of three capacitors namely, C_Σ^P, C_Σ^n and C_j, connected in series. C_T is assigned to the sum of all 2-term products of these three capacitances C_Σ^P, C_Σ^n and C_j.
Now we will find the expression regarding the critical voltage V_c of the tunnel junction. We assume the capacitance of the tunnel junction to be C_j and the remaining part of the circuit has the equivalent capacitance to be C_e, as observed from the point of view of tunnel junction, we can measure the threshold or critical voltage [1,2, 8,9,10] for the tunnel junction as below.

$$V_c = \frac{e}{2(C_j + C_e)} \quad \cdots\cdots (6)$$

$$V_c = \frac{e}{2[C_j + (C_\Sigma^P || C_\Sigma^n)]}$$

$$= \frac{e}{2[C_j + \frac{(C_\Sigma^P) * (C_\Sigma^n)}{(C_\Sigma^P + C_\Sigma^n)}]}$$

$$= \frac{e(C_\Sigma^P + C_\Sigma^n)}{2[C_j * (C_\Sigma^P + C_\Sigma^n) + (C_\Sigma^P) * (C_\Sigma^n)]}$$

$$= \frac{e(C_\Sigma^P + C_\Sigma^n)}{2C_T} \quad \text{.....(7)}$$

When the voltage of the junction is V_j , a tunneling event comes to happen through this tunnel junction if and only if the condition given below is satisfied.

$$|V_j| \geq V_c \quad \text{.....(8)}$$

From this equation it is decided that if the junction voltage is less than the critical voltage i.e. $|V_j| < V_c$, then no tunneling events through the tunnel junction happens. As a consequence, the tunneling circuit keeps its *stable state*.

Theoretically, the thresholds are being integer numbers though it can be a real number. The threshold logic equations for 2-input or 3-input logic AND, OR, NAND and NOR gates are shown in sections from 5 to 9.

3. Buffer

The buffer or inverter [2, 3, 4, 8, 10] depicted in Fig. 2(a) is made up of two single electron transistors (SETs) connected in series. The two input voltages having the same values are directly coupled to the islands of the SET1 and SET2 [6,7] through two capacitors C_1 and C_2 of same values respectively. The islands of each SETs have a size close to 10 nm diameter of gold and their capacitances should be less than 10^{-17} F. The output terminal V_0 is connected to the common channel between SET1 and SET2 and to the ground through a capacitor C_L to put down charging effects.

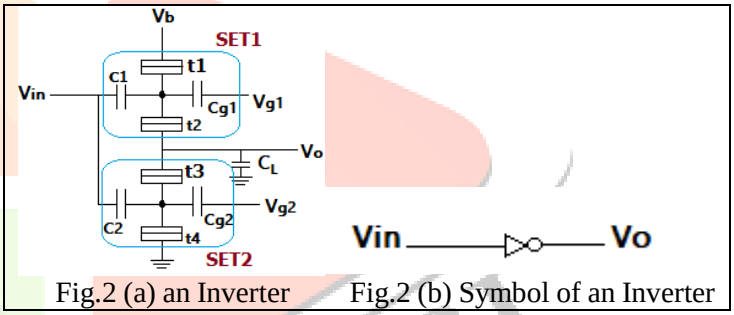
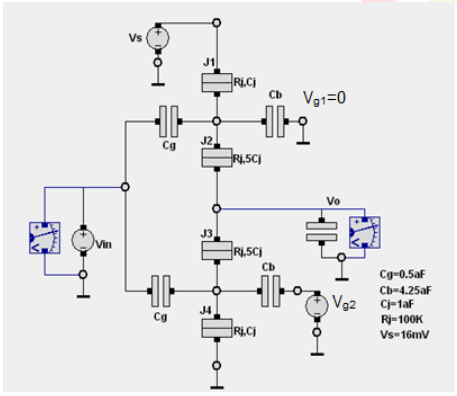
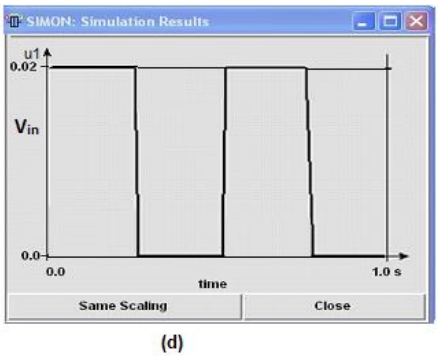


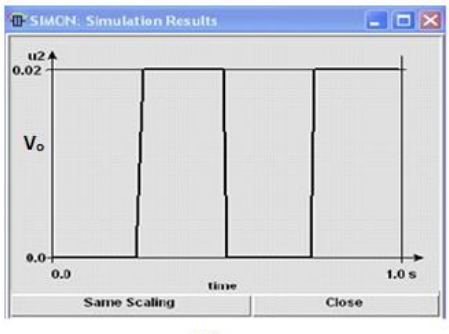
Fig.2 (a) an Inverter Fig.2 (b) Symbol of an Inverter

For the buffer, the parameter values chosen are: V_{g1}=0, V_{g2}=0.1× $\frac{q_e}{C}$, $C_L = 9C$, $t_4 = \frac{1}{10}C$, $t_3 = \frac{1}{2}C$, $t_2 = \frac{1}{2}C$, $t_1 = \frac{1}{10}C$, $C_1 = \frac{1}{2}C$, $C_2 = \frac{1}{2}C$, $C_{g1} = \frac{17}{4}C$ and $C_{g2} = \frac{17}{4}C$, R1 =R2=50KΩ. For simulation purpose, the value of C is taken 1aF.





(d)



(e)

Fig. 2(c) Simulation set of Buffer **Fig.2 Simulation result of Buffer (d) input (e) output**

The operation of the buffer will be described as: - the output V_0 value will be high in case the input voltage V_{in} is low and the V_0 value will be low in case the input voltage is high. To achieve this target, we must set the voltages $for\ V_{g1} = 0\ and\ V_{g2}$ =16mV along with the tuning gate voltages, at present, V_{in} both for SET1 and SET2. SET1 is in conduction mode if V_{in} is set to low and the SET2 is in Coulomb blockade [2, 3, 4, 13, 16, 17]. This results the output voltage V_0 is

linked to V_b and therefore the output voltage becomes high. Coulomb blockade troubles the steady flow of current, as the high voltage (logic 1) is applied to the input terminal(s), it makes shift the induced charge on each of the islands of the two SETs by a fraction of an electron charge and enforces the SET1 in Coulomb blockade and the SET2 in conducting mode. So, the output shifts from high to low (logic 0).

In this work we assume the Boolean logic inputs "0" =0 Volts and logic "1"=$0.1\times \frac{q_e}{C}$.

For simulation and other purposes, will consider that C=1aF and Logic "1"= $0.1\times \frac{1.602\times10^{-19}}{1\times10^{-18}}$=$0.1 \times 1.602 \times 10^{-2}$=$16.02 \times 10^{-3}$ =$16.02 \cong 16$ mV.

4. LOGIC GATES AND LINEAR SEPARATION

Before discussing about different threshold logic gates, one should be known about linear condition of a function. We are trying to make understand how shall we comment about a linear or non-linear function? Let us see the input space for two inputs variables x1 and x2, and how the basic gates are implemented by linear separation. Fig. 3 shows the solution spaces functions of gates AND, OR, NOR and XOR with 4 vertices (combinations 00, 01, 11, 10) as points and green circle-points have the function value 1, and the colorless small circles/bubbles points indicate 0(zero) value for the function. In Fig. 3 (a) and 3(b) and 3(c), we can draw more straight lines (in other words different weights and thresholds) to separate 1-vertices from 0-vertices and thus the functions of AND, OR and NOR can be implemented ,as they are linearly separable, by a single neuron. This is not true for XOR/(XNOR) in Fig. 3(d), since no line can separate the 1-vertices of XOR function from its 0-vertices. So, XOR is not being a linearly separable function. In two dimensions, a line separates points in a plane, whereas a plane is considered as a separator of points in three dimensional space. In general, n-1 dimensional plane (also called hyper plane) is a separator for points in n-dimensions. We shall clarify the linearly separable cases by the following three examples.

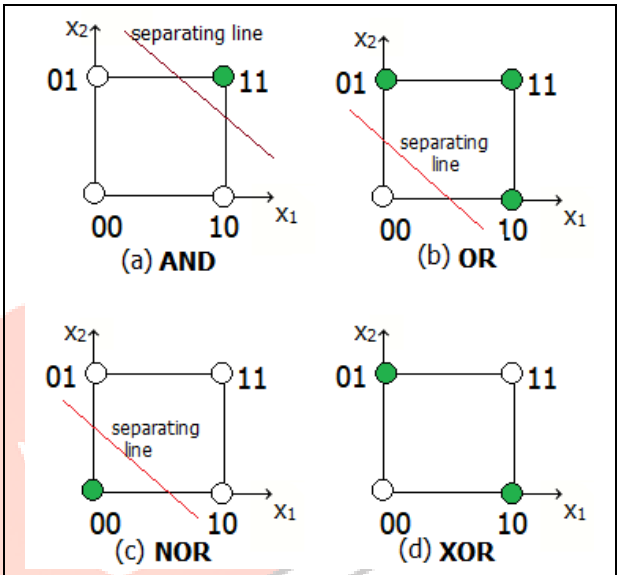


Fig. 3 Solution spaces of two variables x1 and x2 for (a) AND, (b) OR , (c) NOR and (d) XOR gates

Definition 1: A switching function is considered as linearly separable iff all of its 1-vertices (like green colored) can be separated from all of its 0-vertices with the help of a hyper plane. A neuron bearing $n-$ $inputs$ works on $an\ n-dimensional$ space and a single neuron which is capable of implementing any switching function is thought to be linearly separable.

Example 1:

A single neuron can implement a switching function $f(x_1, x_2, x_3) = x_1 + x_2\ x_3 + x_1x_2$, since its 1-vertices can be separated from 0-vertices by a plane as shown in Fig. 4 and its truth table is shown in Table-1.

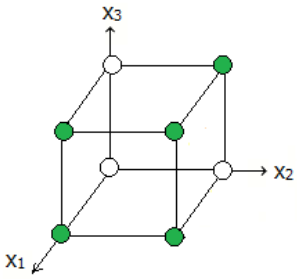


Fig. 4 Space of non-linear threshold logic function

Table-1

x_1	x_2	x_3	f(x_1,x_2,x_3)
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

Example 2:

A single neuron can implement a switching function $f(x_1, x_2, x_3) = x_1 + x_2 + \bar{x}_2\, x_3$, since its 1-vertices can be separated from all 0-vertices by a plane as shown in Fig. 5 and its truth table is shown in Table-2.

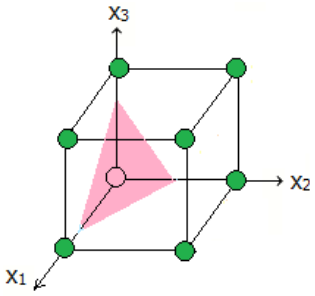


Fig. 5 Space of linear threshold logic function

Table-2

Truth table of $f(x_1, x_2, x_3) = x_1 + x_2 + \bar{x}_2 x_3$,

x_1	x_2	x_3	$f(x_1, x_2, x_3)$
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

Example 3:

A single neuron can implement a switching function $f(x_1, x_2, x_3) = x_1 x_3 + x_2 \bar{x}_3$, since its 1-vertices can be separated from all 0-vertices as shown in Fig. 6 and its truth table is shown in Table-3.

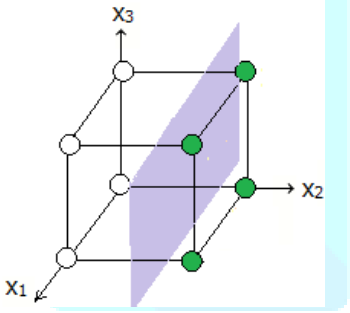


Fig. 6 Space of a linear threshold logic function

Table-3

Truth table of $f(x_1, x_2, x_3) = x_1 x_3 + x_2 \bar{x}_3$

x_1	x_2	x_3	f(x_1, x_2, x_3)
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

5. Threshold logic equation for OR gate

For making the threshold logic gate of an OR gate, we draw the truth table Table-4 of OR gate and compare the weights of variables w_A and w_B of two variables A and B respectively with the threshold value θ.

Table-4

A	B	F(A,B)	θ
0	0	0	$0<\theta$
0	1	1	$w_B \geq \theta$
1	0	1	$w_A \geq \theta$
1	1	1	$w_B + w_A \geq \theta$

For positive logic we assume weights of A and B are 1 each. Then we write the three equations taken from 4th column of Table-4.

$$w_B > \theta \quad \cdots\cdots (9)$$
$$w_A > \theta \quad \cdots\cdots (10)$$
$$w_B + w_A > \theta \quad \cdots\cdots (11)$$

As $0<\theta$, θ must be positive, If we assume w_B=1, w_A=1 and θ=0.5, then the three equations in (9), (10) and (11) are satisfied. Hence the Threshold logic equation for OR gate is given in equation (12) and its corresponding threshold logic gate is drawn in Fig. 5(a)

$$OR(A, B) = sgn\{A + B - 0.5\} \quad \cdots\cdots (12)$$

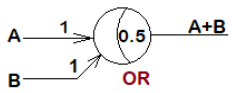
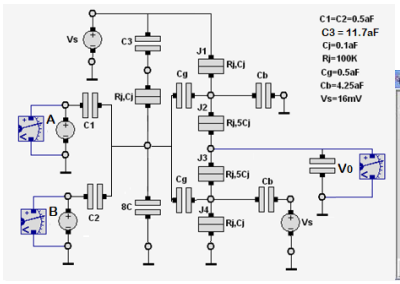


Fig. 5(a) Threshold logic OR gate

For implementing the AND gate we will use the parameters $C_1^n = C_2^n$=0.5aF,$C_3 = 11.7aF$, $C_{b1} = C_{b2} = 4.25aF$, $C_{g1} = C_{g2} = 0.5aF$, $C_L = 9aF, C_0 = 8aF$, R_j=10^5 Ω,$V_s = 16mV$ and accordingly after running the simulator, the output we get is given in Fig. 5(c).



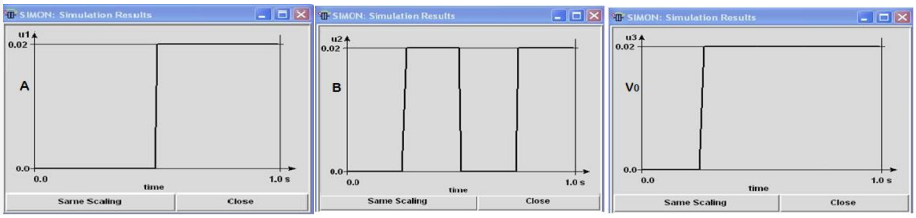


Fig. 5(b) OR gate

Fig. 5(c) simulation result of OR gate

6. 2-input AND gate

For making the threshold logic gate of AND gate, we draw the truth table Table-5 of AND gate and compare the weights of variables w_A and w_B of two variables A and B respectively with the threshold value θ [1,2,3,8].

Table-5

A	B	F(A,B)	θ
0	0	0	$0<\theta$
0	1	0	$w_B < \theta$
1	0	0	$w_A < \theta$
1	1	1	$w_B + w_A \geq \theta$

For positive logic we assume weights of A and B are 1 each. Then from the above four inequalities if we assume w_B=1, w_A=1 and θ=2, then the three inequalities or equations in 4th column in Table-5 are satisfied. Hence the Threshold logic equation for AND gate is given in equation (13) and its corresponding threshold logic gate is drawn in Fig. 6(a)

$$AND(A,B) = sgn\{A + B - 2\} \quad \cdots\cdots (13)$$

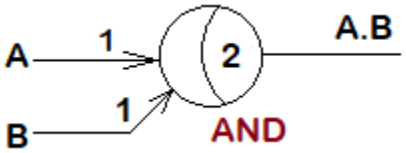
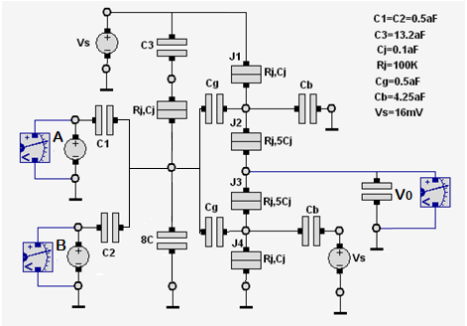


Fig. 6(a) Threshold logic AND gate

For implementing the AND gate we will use the parameters $C_1^n = C_2^n$=0.5aF, $C_{b1} = C_{b2} = 4.25aF$, $C_{g1} = C_{g2} = 0.5aF$, $C_L = 9aF, C_0 = 8aF$, R_j=10^5 Ω in Fig. 6(a) and accordingly after simulation the result we get is given in Fig. 6(c)and the simulation set is shown in Fig. 6(b).



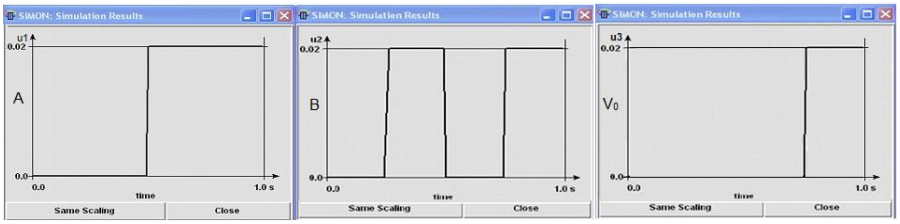


Fig.6 (b) AND Gate

Fig.6 (c) Simulation result of AND gate

7. Threshold logic equation for 3-input AND gate

Table-6

A	B	C	F(A,B,C)=ABC	θ
0	0	0	0	$0< \theta$
0	0	1	0	$\theta > w_C$
0	1	0	0	$\theta > w_B$
0	1	1	0	$\theta > w_B + w_C$
1	0	0	0	$\theta > w_A$
1	0	1	0	$\theta > w_A + w_C$
1	1	0	0	$\theta > w_A + w_B$
1	1	1	1	$w_A + w_B + w_C \geq \theta$

As AND gate is a positive logic we shall assume that all the values of w_A, w_B, w_C and θ are positive. If we take $w_A = 1$, $w_B = 1$,w_C=1 and $\theta = 2.5$ (or any value in the range $2 < \theta \leq 3$), then all the inequality equations in the 5th column are satisfied. So the Threshold logic equation for 3-input AND gate is

$$AND(A,B,C) = sgn\{A + B + C - 2.5\} \ldots\ldots\ldots (14)$$

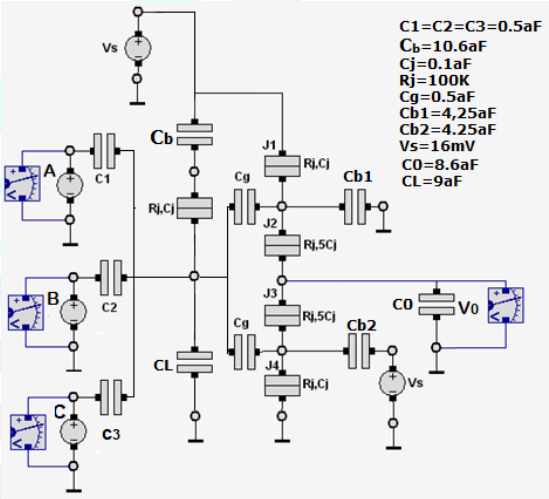
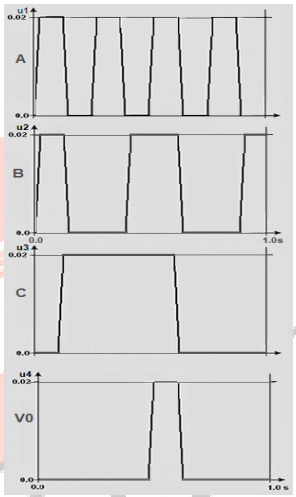


Fig. 6(d) Simulation set of 3-input AND gate



(e) Simulation result of 3-input AND gate

Simulation set of 3-input AND gate is depicted in Fig. 6(d) and the simulation result is shown in Fig. 6(e). As stated earlier, the threshold gates those are derived from the generic threshold gate given in Fig.1 require an output buffer for the correct operation in a network structure. It is well-known that the applied buffer inverts its own input signal, we modify the threshold equation of $AND(A,B,C)$ such that it determines NAND(A,B,C) = $\overline{\text{AND(A,B,C)}}$. So when we combine the result of $\overline{\text{AND(A,B,C)}}$ and a buffer in series, a buffered gate obtained calculates $AND(A,B,C)$. Truth table of 3-input NAND gate and the threshold relationships with the weighted sum is given in Table-7.

$$\text{NAND}=\overline{\text{AND(A,B,C)}} = sgn\{-A - B - C + 2.5\}$$
$$= sgn\{-A - B - C - (-2.5)\} \cdots\cdots\cdots\cdots (15)$$

Table-7
Truth table of Threshold NAND gate

A	B	C	$\{w_A + w_B + w_C\}$	$\theta = -2.5$	F(A,B,C) $= \overline{ABC}$
0	0	0	0	$0 > -2.5$	1
0	0	1	-1	$-1 \geq -2.5$	1
0	1	0	-1	$-1 \geq -2.5$	1
0	1	1	-2	$-2 \geq -2.5$	1
1	0	0	-1	$-1 \geq -2.5$	1
1	0	1	-2	$-2 \geq -2.5$	1
1	1	0	-2	$-2 \geq -2.5$	1
1	1	1	-3	$-3 < -2.5$	0

From the Table-7, it is transparent that equation (15) acts as the equation of a 3-input NAND gate. As a consequence, when we combine this NAND gate and the buffer in series, a 3-input AND gate providing correct output is obtained.

8. Threshold logic equation for OR and NOR Gate

We are interested in implementing a 2- input NOR gate to be derived from the generic threshold gate given in Fig.1. To implement the 2- input NOR gate, we require a threshold logic 2-input OR gate which will be connected with a buffer in series. Comparing with the generic threshold gate based general equation given in equation (2), we can think of a gate of two variables A and B of function F(A,B) as.

$$F(A,B) = sgn\{w_A . A + w_B . B - \theta\} \cdots\cdots\cdots\cdots (16)$$

For a positive logic we should assume that all the values of w_A, w_B and θ are positive. If we take $w_A = 1, w_B = 1$ and θ =0.5 (or any value in the range $0 < \theta \leq 1$), then the above equation becomes

$$F(A,B) = sgn\{A + B - 0.5\} \cdots\cdots\cdots\cdots (17).$$

And if we construct the truth table of it we get the Table-8.

Table-8

A	B	$\{A, w_A + B.w_B\}$	$\theta = 0.5$	$F(A,B)$
0	0	0	$0 < 0.5$	0
0	1	1	$1 \geq 0.5$	1
1	0	1	$1 \geq 0.5$	1
1	1	2	$2 \geq 0.5$	1

The Table -8 satisfies the condition of an OR gate, so equation (17) is written as

$$OR(A,B) = sgn\{A + B - 0.5\} \cdots\cdots\cdots (18)$$

As already discussed that buffer inverts itself i.e., inverts its own input signal, we modify the threshold equation of OR(A,B) such that it determines NOR(A,B). So when we combine the result of OR(A,B) with a buffer or inverter, a new threshold logic buffered gate we get that calculates the value of NOR (A, B) and the equation of the NOR logic gate will be

$$NOR(A,B) = \overline{OR(A,B)}$$
$$= sgn\{-A - B - (-0.5)\} \cdots\cdots (19)$$

Here the value of the threshold voltage -0.5 may be any value in the range of $-1 > \theta \geq 0$.

Table-9

A	B	$\{W_A + W_B\}$	$\theta = -0.5$	$F(A,B)$
0	0	0	$0 \geq -0.5$	1
0	1	-1	$-1 < -0.5$	0
1	0	-1	$-1 < -0.5$	0
1	1	-2	$-2 < -0.5$	0

From the Table -9 it is observed that F (A, B) satisfies all the conditions of an NOR gate, so the equation (19) we derived is correct.

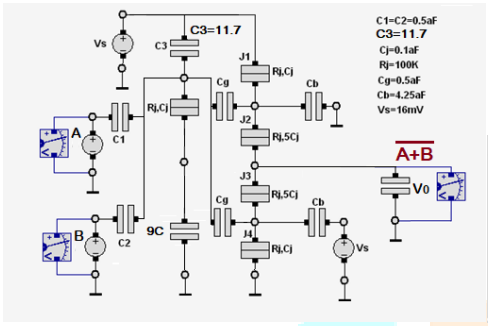
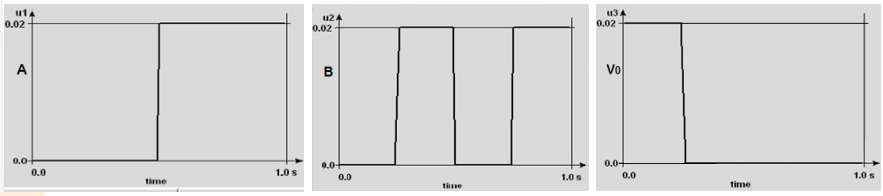


Fig 7(a) TLG based NOR gate



Fig, 7(b) Simulation result of NOR gate

For implementing the buffered Boolean logic NOR gate we will use the parameters logic input "0"=0V, logic "1" = 16mV, $C_1^n = C_2^n$=0.5aF,$C_3 = 11.7aF$, $C_b = C_{b1} = C_{b2} = 4.25aF$, $C_{g1} = C_{g2} = 0.5aF$, $C_L = 9aF, C_0 = 9aF$, R_j=10^5 Ω, $V_s = V_b = 16mV$ and accordingly after simulation the result we get is given in Fig. 7(b).

9. XOR Gate

The logic function of XOR gate is defined as Y=A.$\bar{B}$ +$\bar{A}$.B, where A and B are two variables. Space plot diagram of Y in 2D space is shown in Fig. 8. From this, we observe that no linear separating line that is separating the green and colorless bubbles can be drawn. So the Boolean logic function is not linearly separable. Therefore we would not be able to draw a threshold logic gate that represents the equation Y=A.$\bar{B}$ +$\bar{A}$.B.

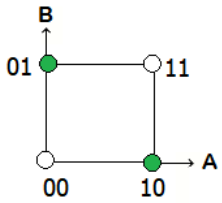


Fig.8 Space plot of Y=A.$\bar{B}$ +$\bar{A}$.B

Now for representing the Boolean function Y=A.$\bar{B}$ +$\bar{A}$.B by a threshold logic gate first we express P = (A.$\bar{B}$) with the help of threshold gate-based equation as given in equation (13).

P = *sgn* {A+ $\bar{B}$−2}····· (20)

As we know B + $\bar{B}$ =1 or $\bar{B}$= − B + 1, the equation (20) can be written as equation (21).

P = *sgn* { A − B − (1) }····· (21)

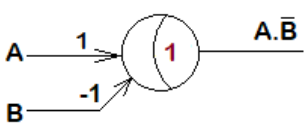


Fig. 9(a) Threshold gate of P = A.$\bar{B}$

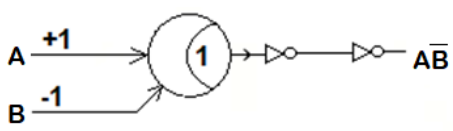


Fig. 9(b) Threshold gate of A.$\bar{B}$ using double buffers

Table-10
Truth table of equation (21)

A	B	P
0	0	0
0	1	0
1	0	1
1	1	0

We can modify the threshold equation of P as we know that the buffer inverts its input signal, so that it calculates $\bar{P}$=sgn $\{-A+B-(-0.5)\}$ and its corresponding truth table is given in Table-11.

Table-11
Truth table of $\bar{P}$

A	B	$\bar{P}$
0	0	1
0	1	1
1	0	0
1	1	1

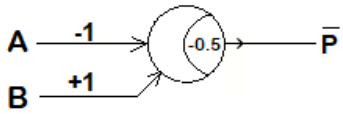


Fig. 10 Threshold logic gate of $\bar{P}$

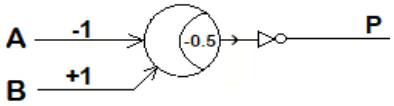


Fig. 11 Complement of Fig.10

Now we are to express the Boolean expression Y=A.$\bar{B}$ +$\bar{A}$.B. We assume P= $\bar{A}$.B, So Y=A.$\bar{B}$ +$\bar{A}$.B = P + $\bar{A}$.B

$$Y = P + \bar{A}.B \quad \cdots\cdots (22)$$

For finding out the threshold gate logic of equation (22), we draw the truth Table-12 with the assistance of Table-11 and from which we solve the equations to get the weight values.

$0<\theta$…… (23)
$W_2 \geq \theta$…… (24)
$W_1 + W_3 \geq \theta$…… (25)
$W_1 + W_2 < \theta$…… (26)

Table-12

A	B	P	Y	θ
0	0	0	0	$0<\theta$
0	1	0	1	$W_2 \geq \theta$
1	0	1	1	$W_1 + W_3 \geq \theta$
1	1	0	0	$W_1 + W_2 < \theta$

From the equation (23), we have θ is greater than 0, so it may be 1 or 2 or 3.
From equation (24) it is transparent that W_2 must be equal to or greater than θ. For minimum value condition we can take as:

$$\theta = W_2 = 1 \quad \cdots\cdots (27)$$

In equation (26) if we put $W_1 = -1$ and W_2= 1 then the equation is satisfied, so we take $W_3 = 2$
Next in equation (28) if we put $W_2 = -1$, $W_1 = 2$ and θ =1 then it is satisfied. So a solution set is $\{ W_1 , W_2 , W_3 ; \theta \}$ = { -1, 1, 2 ; 1}

$$Y=\text{sgn}\{W_1.A + W_2.B + W_3.P - (\theta)\} \quad \cdots\cdots (28)$$

Hence the Threshold equation for the Y or an for XOR is

$$Y = \text{sgn}\{-A + B + 2P - (1)\} \quad \cdots\cdots (29)$$

And its corresponding Threshold logic gate is depicted in Fig. 12.

For correct operation, we are to apply a buffer in series to obtain an XNOR which is shown in Fig. 12. If we again add another buffer in series we obtain an XOR gate as shown in Fig. 12 as well.

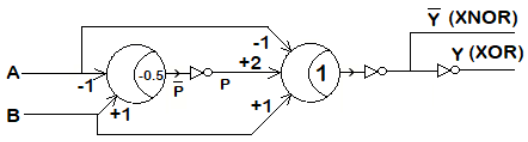


Fig. 12 XOR/ XNOR gate of two inputs

10. Regarding Full Adder

Sum of a Full adder is $S=a\oplus b\oplus c_0=\bar{a}bc_0+a\bar{b}c_0+ab\bar{c}_0+abc_0$ which is not linearly separable, because the solution space can't be separated by a plane into two parts of green-colored and colorless circle points indicating 1-points and 0-points respectively as shown in Fig. 13(a) below.

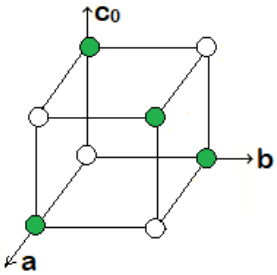
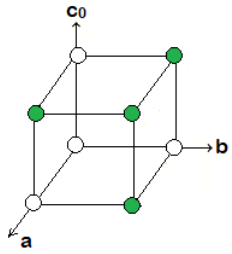


Fig.13 (a) space plot diagram of S



Fig, 13(b) space plot of $c_1=ab+bc_0+c_0a$

From the carry equation $C_1=ab+bc_0+c_0a$ if we draw the space solution diagram, it would be clear that the 1-points are linearly separable from 0-points. Therefore the C_1 can be turned into a linear threshold gate. Consider the equation below,

C_1=sgn {a+b+C_0-θ}.....(30)

Table-13

	Inputs			output	Threshold Condition
Sl. No.	a	b	c_0	c_1	θ
(1)	0	0	0	0	0< θ
(2)	0	0	1	0	w3< θ
(3)	0	1	0	0	w2< θ
(4)	0	1	1	1	w2+w3≥ θ
(5)	1	0	0	0	w1< θ
(6)	1	0	1	1	w1+w3≥ θ
(7)	1	1	0	1	w1+w2≥ θ
(8)	1	1	1	1	w1+w2+w3≥ θ

As C_1 is a positive logic, we must take the weights as low integer value i.e., w1=w2=w3=1. From the row (1) of the Table-13, the value of θ must be positive. Since we have taken w1=w2=w3=1 and w1=w2=w3< θ (from the rows (2), (3) and (5)), we take the minimum integer value of θ equal to 2. After analyzing logically, we have taken the parameter values as: w1=w2=w3=1 and θ = 2. Now, if we put these values in the conditional equations in the 6th column in Table-13, all the equations are satisfied. So, one solution set of the parameters is { w1,w2,w3; θ} ={1,1,1;2}. So the threshold equation of Carry c_1 will be

$C_1= sgn\{a+b+c_0-2\}$.....(30a)

According to this solution set, the threshold logic gate can be drawn in Fig.14.

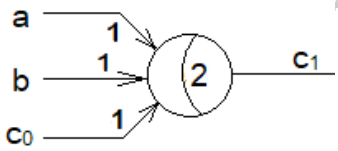


Fig. 14 TLG diagram of C_1

Now for calculating the threshold logic gate of Full adder we shall consider the sum equation $S=\bar{a}bc_0+ a\bar{b}c_0+ ab\bar{c}_0+ abc_0$, and carry equation $c_1=ab+bc_0+c_0a$ and construct the truth table of sum S, taking the four variables a, b, c_0 and c_1. The truth table is given in Table-14.

From the row (1) in Table-14, we have the value of θ to be positive. Since S is a positive logic, so its coefficient values would be positive. Therefore, we can take them as lowest integers i.e., w1=w2=w3=1. Under these assumptions and comparing the equation in rows (2), (3) and (5) from the Table-14, we should take the value of θ as minimum positive integer as 1, i.e., θ=1. After that, if we put the value of w4= −2, the all the conditional equations are satisfied. So, one solution set is $\{w1,w2,w3,w4;\ \theta\} = \{1,1,1,-2;1\}$. So the threshold equation of Sum (S) will be

$S=sgn\{a+b+c_0-2c_1-1\}$.....(30b)

In accordance with the solution set, the threshold logic gate is drawn in Fig.15.

Table-14

	Inputs			out put	Sum	Threshold Condition
Sl. No.	**a**	**b**	**c_0**	**c_1**	**S**	**θ**
(1)	0	0	0	0	0	$0< \theta$
(2)	0	0	1	0	1	$w3\geq \theta$
(3)	0	1	0	0	1	$w2\geq \theta$
(4)	0	1	1	1	0	$w2+w3+w4 < \theta$
(5)	1	0	0	0	1	$w1\geq \theta$
(6)	1	0	1	1	0	$w1+w3+w4 < \theta$
(7)	1	1	0	1	0	$w1+w2+w4 < \theta$
(8)	1	1	1	1	1	$w1+w2+w3+w4 \geq \theta$

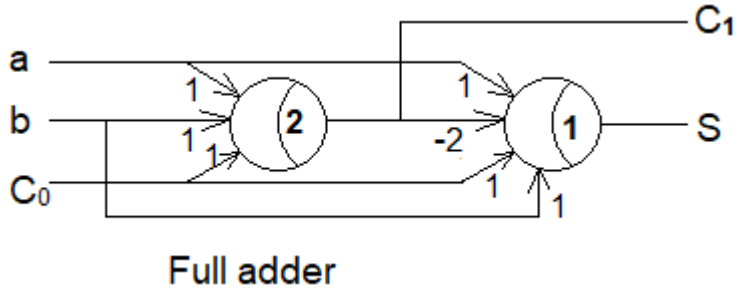


Fig. 15 Full Adder Circuit

Theoretically the above circuit in Fig. 15 is correct at 0K, but when temperature fluctuates then the correctness of the output stands on a question mark. For correct operation, one buffer circuit must be appended in series with every Threshold Logic Gate. For doing so, first the carry of the Full adder is taken in our justification. Complement of threshold logic of carry C_1 is thought of in equation (31).

$\bar{C}_1$ =sgn {w1.a+ w2.b+ w3.c_0-θ}…….(31)

From the Table-15, it would be clear that if we take the value of θ equal $to\ -1.5$ and w1= −1, w2= −1 and w3= −1 then all the conditions in 5th column of the Table-15 are satisfied. Hence the equation turns to the following.

$\bar{C}_1$ =sgn {$-a-b-c_0-(-1.5)$}……. (32)

As per the equation (32), we can draw the threshold logic gate with a buffer to get C_1 shown in Fig.16.

Table-15

Inputs			output	Threshold Condition
a	b	c_0	$\bar{c}_1$	
0	0	0	1	$0\geq \theta$
0	0	1	1	$w3\geq \theta$
0	1	0	1	$w2\geq \theta$
0	1	1	0	$w2+w3< \theta$
1	0	0	1	$w1\geq \theta$
1	0	1	0	$w1+w3< \theta$
1	1	0	0	$w1+w2< \theta$
1	1	1	0	$w1+w2+w3< \theta$

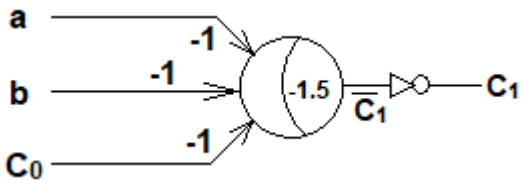


Fig.16 Carry of a full Adder using buffer

Now we are to find out the complement of sum $S = sgn\ \{a + b + c_0 - 2c_1 - 1\}$.

Consider $\bar{S}$=sgn {w1.a+w2.b+w3.c_0+w4.c_1-(θ)}

Table-16
Truth table of sum of a full adder

Inputs				sum	Threshold Condition
a	b	c_0	c_1	$\bar{S}$	θ
0	0	0	0	1	$0\geq \theta$
0	0	1	0	0	$w3< \theta$
0	1	0	0	0	$w2< \theta$
0	1	1	1	1	$w2+w3+w4 \geq \theta$
1	0	0	0	0	$w1< \theta$
1	0	1	1	1	$w1+w3+w4 \geq \theta$
1	1	0	1	1	$w1+w2+w4 \geq \theta$
1	1	1	1	0	$w1+w2+w3+w4 < \theta$

$\bar{S} = sgn\{w1.a + w2.b + w3.c_0 + w4.c_1 - (\theta)\}$ ..... (33)

After solving the conditional equations in 6th column of Table-16, we obtain a solution set as $\{w1, w2, w3, w4;\ \theta\} = \{-1, -1, -1, +2;\ -0.5\}$ for the equation (33). So the complement of Sum equation will be

$\bar{S} = sgn\{-a - b - c_0 + 2c1 - (-0.5)\}$ ..... (34)

As per the equation (34), we can draw the threshold logic gate with a buffer to get S.

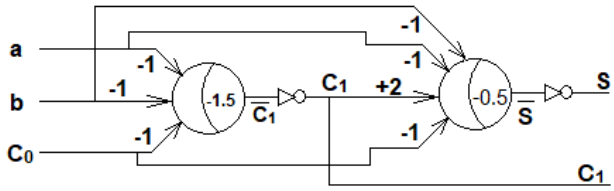


Fig. 17 Full Adder using TLG and Buffer

The full adder based on threshold gate and buffer drawn in Fig. 17 is more stable and correct in comparison with Fig.15. Because as soon as we would like to draw a circuit based on TLG we must include a buffer to that TLG for getting correct answer.

When we are going to simulate the full adder we choose the parameters [A linear 7, 8] as below.

The threshold logic gate input logic "0"=0V, logic "1" = 16mV, C=1aF, $C_1^P = C_2^P = C_3^p = C_1^n$ =0.5aF, $C_b = 18.5aF$, $C_j = 0.25aF$, $C_L = 9aF$, $C_0 = 9.5aF$, R_j=10^5 Ω, . $V_b = 12.8mV$.

Simulation set of a Full adder and the simulation results (using SIMON) of the Full adder is depicted in Fig. 18(a) and 18(b),(c), (d), (e) and (f) respectively.

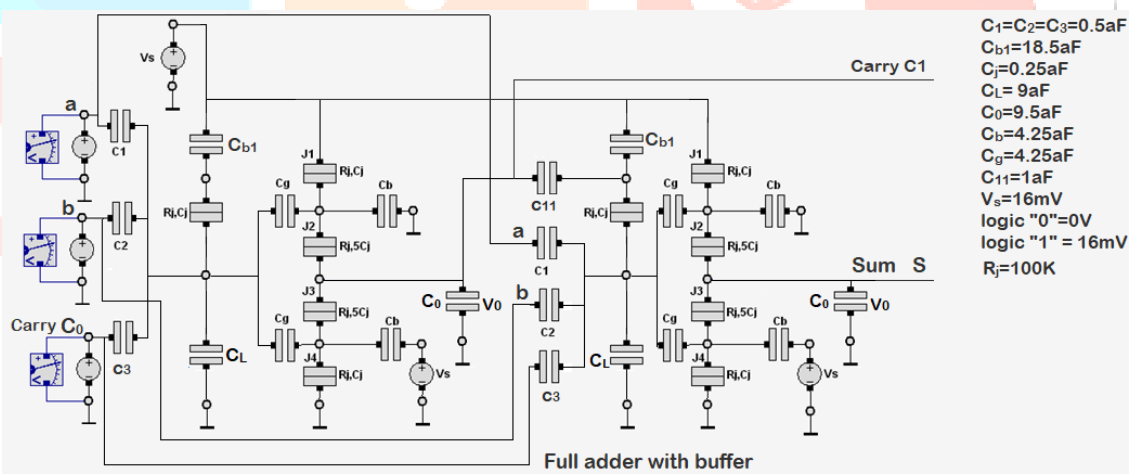


Fig. 18(a) Full adder (correct) with buffer(s)

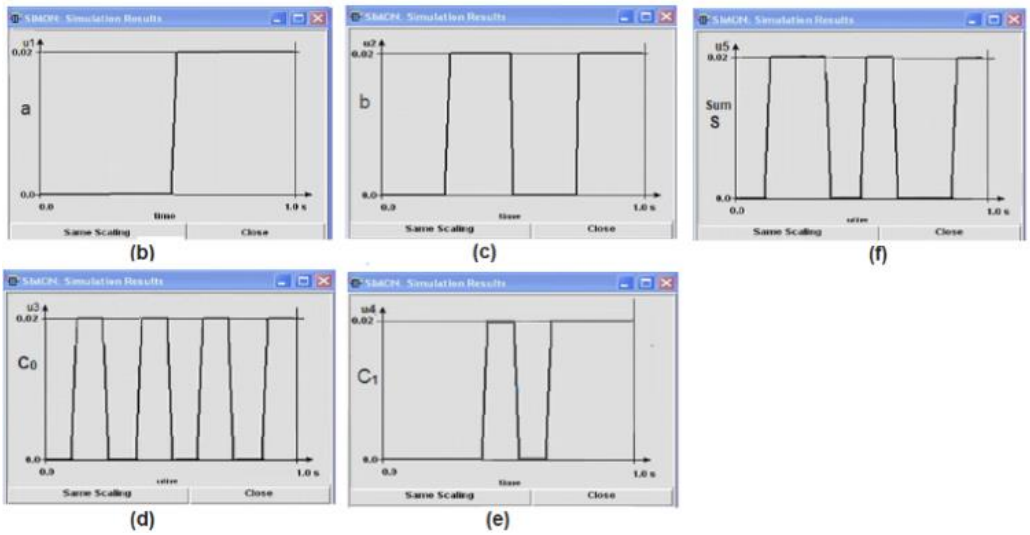


Fig. 18(b) Simulation result of variable a, (c) of b,
(d) of c_0, (e) of c_1 and (f) of sum S of Full adder

11. BCD to excess-3

The availability of a large variety of codes in the digital system for the same discrete elements of information results in the use of different codes by different digital systems. It is sometimes essential to use the output of one system as the input to another for reasonable purpose. A converter or its circuit must be inserted between the two systems if each uses different codes for the same information. The circuit that makes the two different systems compatible even though each uses a different binary code is defined as a code converter. We are interested in making a code converter named BCD-to-excess-3 by using the threshold logic gates (TLGs).

Table-17

Decimal	BCD Code				Excess-3			
	A	B	C	D	w	x	y	z
(0)	'0'	0	0	0	0	0	1	1
(1)	0	0	0	1	0	1	0	0
(2)	0	0	1	0	0	1	0	1
(3)	0	0	1	1	0	1	1	0
(4)	0	1	0	0	0	1	1	1
(5)	0	1	0	1	1	0	0	0
(6)	0	1	1	0	1	0	0	1
(7)	0	1	1	1	1	0	1	0
(8)	1	0	0	0	1	0	1	1
(9)	1	0	0	1	1	1	0	0

w=A+BC+BD=A+B(C+D) …….…….. (35)
x=B'C+B'D+BC'D'=B'(C+D) + BC'D'
=B'(C+D) + B(C+D)'=B **XOR** (C+D) …… (36)
y=CD+C'D'= C **XNOR** D……...…….(37)
z=D'……...(38)

For the equations (35) and (36) we assign P=(C+D) then the two equations become

$w = A + BC + BD = A + BP$ …….. (39)
x=B'(C+D) + B(C+D)'=B **XOR** P …… (40)

Now equation (39) is linearly separable, but equations (37) and (40) are not. Now we are taking into consideration w=A+BP which is linearly separable as the five green bubble can be separated from the other bubbles by a plane shown in space solution diagram in Fig. 19. Where green bubbles indicate the values of 1 of w=A+BP.

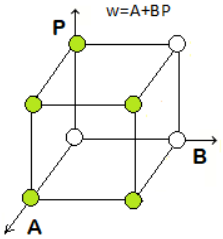


Fig.19

Table-18

A	B	P	w
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

Now we can try to find out the threshold equation of w=A+BC+BD=A+B(C+D). Noted that C+D is an OR Boolean equation and whose threshold logic gate is already shown in Fig. 5(a) yet we are showing it again only changing the variables below in Fig. 19(a).

Fig. 19(a) TLG of P= C+D

Let us assume the threshold logic equation of w is:

$w = sgn\{w1.A + w2.B + w3.P - (\theta)\}$ ..... (41)

Table-18

Inputs			Output A+BP	Threshold Condition
A	B	P	w	
0	0	0	0	$0 < \theta$
0	0	1	0	$w3 < \theta$
0	1	0	0	$w2 < \theta$
0	1	1	1	$w2+w3 \geq \theta$
1	0	0	1	$w1 \geq \theta$
1	0	1	1	$w1+w3 \geq \theta$
1	1	0	1	$w1+w2 \geq \theta$
1	1	1	1	$w1+w2+w3 \geq \theta$

After solving the conditional equations in 5th column of Table-18, we obtain a solution set as {w1, w2, w3; θ } = {4, 3, 2; 3.5} for the equation (41). So the threshold equation of 'w' would be

w=sgn {4A + 3B+ 2P-(3.5)}..... (42)

According to the equation (42) the threshold logic gate is drawn in Fig. 20.

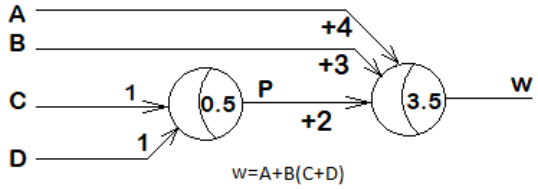


Fig. 20 Threshold gate of w=A+B(C+D)

Till now, we have drawn the threshold logic gate of w=A+B(C+D). So the second step of drawing the Threshold logic gate of the equation (36) is described below. We observe that x is an XOR function of B and (C+D). (C+D) is an OR function. Therefore, Two threshold logic gates (i) OR and (ii) XOR will be applied in series to find out the value of equation (36). Explanation of an OR gate and that of an XOR gate have been given in section (5) and (9) respectively. Equation of OR gate is given in equation (12). The functional threshold gate of equation (36) i.e,. x =B'(C+D) +

B(C+D)'=B **XOR** (C+D) is depicted in Fig.21 using (i) the complement of OR appended with a buffer and (ii) XOR connected with two buffers in series for correct operation.

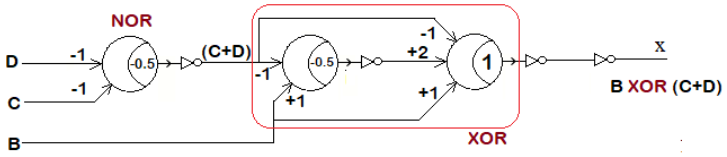


Fig.21 Threshold logic gate of x = B **XOR** (C+D)

Our third step is to implement of y = CD+$\overline{C}\overline{D}$ = (C **XNOR** D) in equation (37). As Y is an Exclusive-NOR gate which has already been discussed in section-9 and the circuit diagram is provided in Fig. 12. So with the help of Fig.12, we have drawn the threshold equation of y = CD+$\overline{CD}$ in Fig. 22 below.

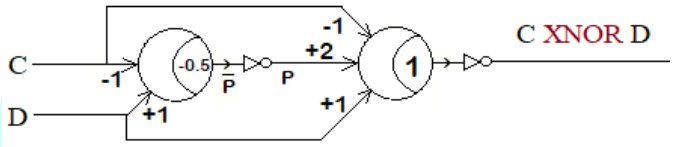
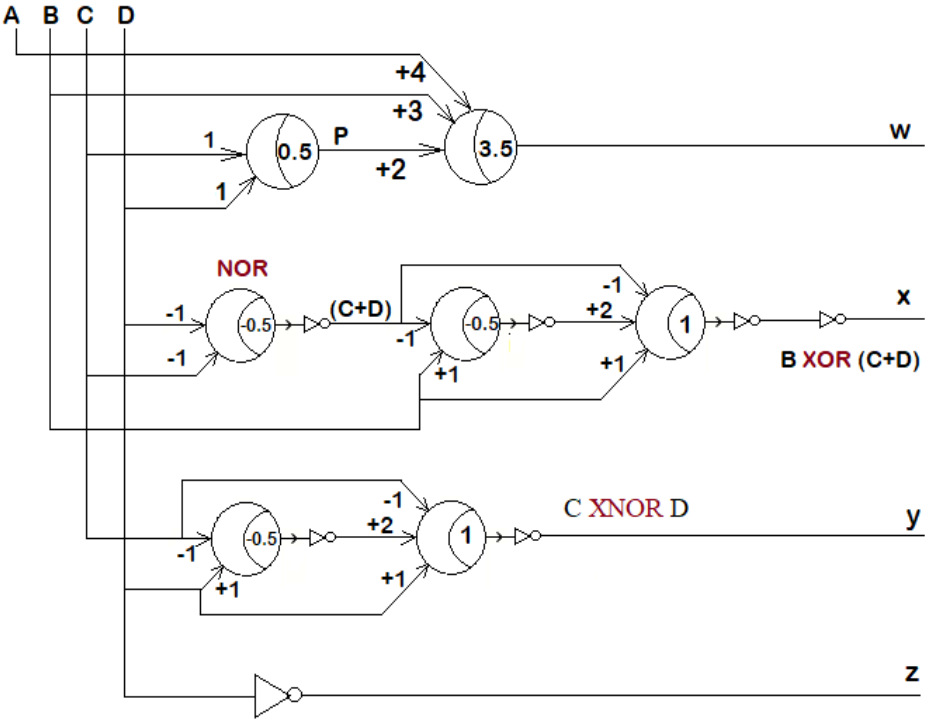


Fig. 22 Threshold Gate of y=CD+C'D'

And the last step is to do the complement of D i.e.,z=D'. To complement 'D' only a buffer is sufficient and a buffer is drawn in Fig. 2(a) or 2(c). The buffer is represented by a symbol  is shown in Fig. 2(b). The Threshold logic gate of a BCD-to-Excess-3 and its simulation result is shown in Fig.23(a) and 23(b) respectively.



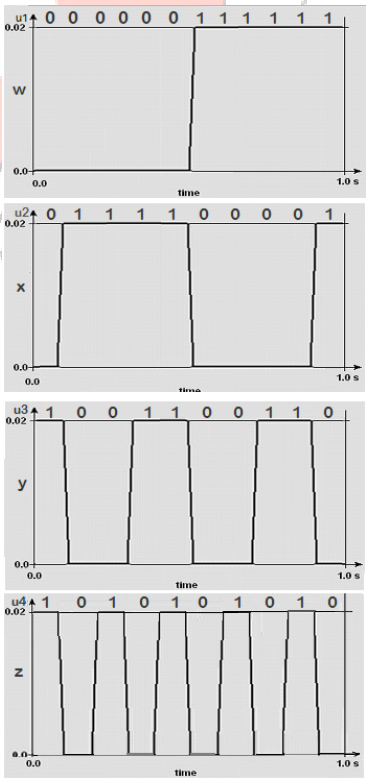


Fig. 23(a) Threshold logic gate based BCD-to-Excess-3 Converter

Fig. 23(b) simulated result

12. Discussion

We have discussed about different logic gates, full adder and a converter based on the LTG gates. We are to observe about their speed which is slow or fast. For calculating the processing delay of any logic gates, we should involve critical voltage V_c given in equations (6) and (7), as well as the tunnel junction capacitance C_j . However, assuming the atmosphere temperature at $T = 0K$, the switching/processing delay of a logic gate can be calculated with the help of the approaches [8, 9].

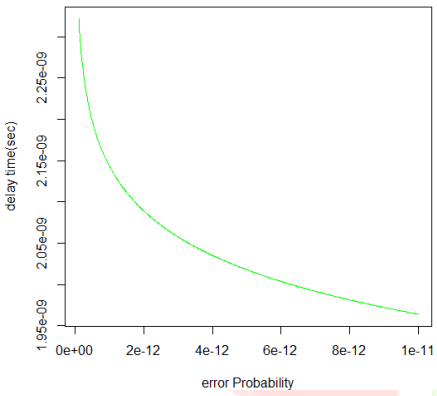
Delay = $-(e|\ln(P_{error})|R_t) / (|V_j| - V_c)$ ..... (43)

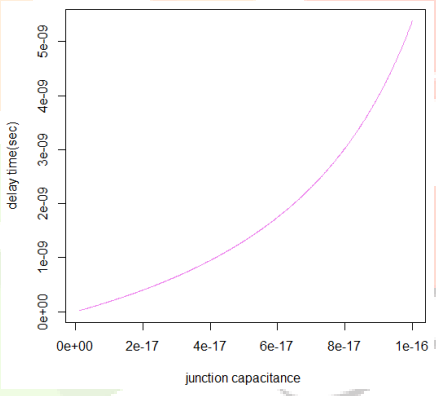
where V_j is the junction voltage and V_c is the critical threshold voltage and R_t being the junction resistance.

The switching will happen whenever the critical voltage V_c has the value less than the tunnel junction voltage V_j, i.e., $V_c < |V_j|$, but very near to it. This happens when V_{in1} is logic 1, resulting V_j =11.8mV for the case of a 2-input NOR gate in Fig-7(a), the critical voltage of the tunnel junction voltage V_c is 11.58mV. Given that the probability of error change P_{error} =10^{-12} and tunnel resistsnce $R_t = 10^5\Omega$. After calculation we get a gate delay equal to $0.07281|\ln(P_{error})|$ns = 1.675 ns. In this manner, we can calculate the circuit delays written in Table-19. Whenever an electron passes through the tunnel junction, the amount of total energy in the circuit changes after the tunneling events. The difference between the energy levels before and after the tunneling event is found to be

$$\Delta E = E_{before\ tunnel} - E_{after\ tunnel} = -e(V_c - |V_j|) \text{ ..... (44)}$$

and it is the amount of switching energy being consumed when a tunnel event occurs in the tunneling circuit.
We have drawn curves as to the switching delay as a function of the switching error probability in Fig. 24(a) and the switching delay as a function of the unit capacitance C shown in Fig. 24(b).





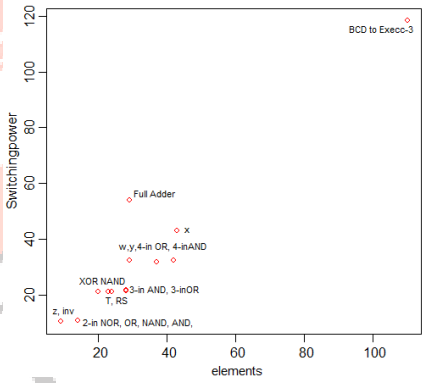


Fig. 24(a) Delay vs. Error Probability Fig. 24(b) Delay Vs. capacitance Fig. 24(c) Switching energy vs. elements

We have counted the element numbers for each and every case of gates or circuits, their switching delays, and switching energy consumptions for the corresponding LTGs (using the same methodology as adopted for the Boolean gates). All the calculated parameters are shown in tabular form in Table-19.
The switching energy vs. elements diagram regarding our present LTG based circuits is shown in Fig. 24(c).
Next, we have calculated those parameters for all gates, Full adder and BCD–to−Excess-3 counter and those are presented in Table-20.

The processing delays for different gates and circuits are different. For 2-input OR switching delay is $0.062|\ln(P_{error})|$ ns, for 3-input AND gate it is $0.104|\ln(P_{error})|$ ns, for full adder it is $0.134|\ln(P_{error})|$ ns ,and for BCD-to-Excess-3 it is $0.206|\ln(P_{error})|$ ns. Given that the value of P_{error} equals to 10^{-12}, so the time after which the 1st output of the bit BCD–to−Excess-3 counter will fan out is $0.206|\ln(P_{error})|$ ns=5.69ns. i.e., after every 5.69 ns, the next output bit will be taken from the counter. Therefore clock time/duration of the clock signal should be more than or equal to 5.69 ns provided synchronization is essential. In this situation, the speed of the BCD–to−Excess-3 counter will be $1/5.69ns$ = 1.75GHz.

Table-19

Gate/Device	elements	Delay	Switching Energy
inverter	09 elements	0.022|ln(P_{error}) | ns	10.4 meV
2-input NOR	14 elements	0.072|ln(P_{error})| ns	10.7 meV
2-input OR	14 elements	0.062|ln(P_{error})| ns	10.8 meV
2-input NAND	14 elements	0.080|ln(P_{error}) |ns	10.7 meV
2-input AND	14 elements	0.062|ln(P_{error}) |ns	10.8 meV
3-input AND	28 elements	0.104|ln(P_{error}) | ns	21.6 meV
3-input NAND	28 elements	0.072|ln(P_{error}) | ns	21.4 meV
2-input XOR	20 elements	0.102|ln(P_{error})| ns	21.2 meV
3-input OR	28 elements	0.104|ln(P_{error}) | ns	21.6 meV
4-input OR	42 elements	0.104|ln(P_{error}) | ns	32.4 meV
4-input AND	42 elements	0.104|ln(P_{error}) | ns	32.4 meV
RS Flip-flop	24 elements	0.082|ln(P_{error}) | ns	21.2 meV
T Flip-flop	23 elements	0.082|ln(P_{error}) | ns	21.1 meV
Trig.T Flip-flop	37 elements	0.144|ln(P_{error}) | ns	31.9 meV
Carry C1	14 elements	0.062|ln(P_{error})| ns	10.8 meV
Full adder	29 elements	0.134|ln(P_{error})| ns	54.0 meV
w	29 elements	0.134|ln(P_{error})| ns	32.4 meV
x	43 elements	0.206|ln(P_{error})| ns	43.1 meV
y	29 elements	0.134|ln(P_{error})| ns	32.4 meV
z	9 elements	0.022|ln(P_{error})| ns	10.4 meV
BCD-to-Excess-3	110 elements	0.206|ln(P_{error})| ns	118.3 meV

We are interested in comparing the circuit delays of CMOS, SET-based and LTG-based. The processing delay or switching delay for a CMOS logic gate like AND, NAND, NOR, XOR is 12ns [18, 19], on the other hand the time required for tunneling through a single electron transistor (SET)[6,7] is approximately 4ns [2,3, 4, 5, 18, 19].

Table-20
Switching delays of SET and LTG

Gate/Device	SET-based delay	LTG-based delay
inverter	8	0.60ns
2-input NOR	4	1.67ns
2-input OR	4	1.71ns
2-input NAND	4	2.21ns
2-input AND	4	1.71ns
3-input AND	8	2.87ns
3-input NAND	8	1.98ns
2-input XOR	4	2.81ns
3-input OR	8	2.87ns
4-input OR	12	2.87ns
4-input AND	12	2.87ns
RS Flip-flop	8	2.26ns
T Flip-flop	8	2,26ns
Trig.T Flip-flop	12	3.98ns
Carry C1	4	1.71 ns
Full adder	8	3.70 ns
w	8	3.70 ns
x	12	5.69 ns
y	8	3.70 ns
z	4	0.60 ns
BCD-to-Excess-3	12	5.69 ns

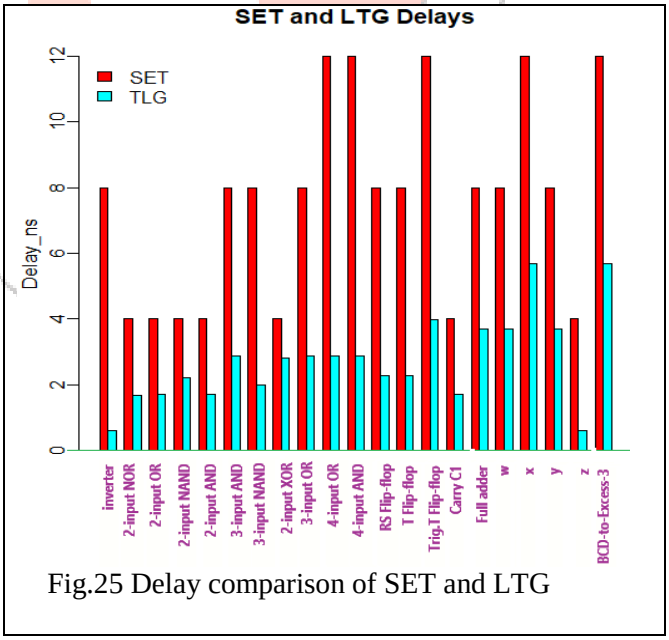


Fig.25 Delay comparison of SET and LTG

The XOR gate using conventional logic circuits needs 16 transistors, whereas the same function can be implemented with the help of 2 TLGs and 2 buffers[1,2, 3, 5, 8, 10] i.e. number of nodes can be reduced to 2 instead of 16.

It is considered that that the error probability is 10^{-12} then the delay for the 3-input OR gate will be 2.87ns and similarly the delays for the other gates can be calculated and are all shown in Table-20. It is clear to us that the LTG based circuit is faster than the SET based circuit when P_{error}=10^{-12}. The comparison of delays for SET and LTG gate based circuits is represented by a bar diagram depicted in Fig.25.

13. Conclusion

We have discussed about how an electron tunnels through a single electron transistor and an inverter. A generic Linear Threshold logic Gate implementation has been discussed about its construction and from which we have been able to derive a family of logic gates like AND, NAND, OR.XOR, Full Adder etc. All the gates along with Full Adder have been implemented and are verified by means of simulation using SIMON. The number of elements requiring for logic gates, and other circuits, their processing delays, power consumed by them are given in tabular form and their related curves or bar diagram are also given in the adjacent figures. By dint of threshold logic equation all the LTG gates have been depicted in due places. In single electron tunneling technology, we have observed that the threshold logic gates are at least 2-times faster than SET based logic gates. Naturally, the temperature of the atmosphere should be kept at 0K in real operation.

14. REFERENCES

[1] Anup Kumar Biswas, "Application of single electron threshold logic gates and memory elements to an up-down Counter" International Journal of Creative Research Thoughts (IJCRT) | Volume 9, Issue 6 June 2021 | ISSN: 2320-2882

[2] Anup Kumar Biswas, -"Implementation of A 4n-Bit Comparator based on IC Type 74L85 using Linear Threshold Gate Tunneling Technology" International Journal of Engineering Research & Technology (IJERT), Vol. 10 Issue 05, May-2021 pp.299-310, ISSN: 2278-0181

[3] Anup Kumar Biswas, "State Transition Diagram for A Pipeline Unit based on Single Electron Tunneling" International Journal of Engineering Research & Technology (IJERT) Vol. 10 Issue 04, April-2021pp.325-336, ISSN: 2278-0181

[4] Anup Kumar Biswas, "Design of A Pipeline for A Fixed-Point Multiplication using Single Electron Tunneling Technology", International Journal of Engineering Research & Technology (IJERT), Vol. 10 Issue 04, April-2021 pp. 86-98, ISSN: 2278-0181

[5] Souvik Sarkar1, Anup Kumar Biswas2, Ankush Ghosh1, Subir Kumar Sarkar1 "Single electron based binary multipliers with overflow detection", International Journal of Engineering, Science and Technology Vol. 1, No. 1, 2009, pp. 61-73

[6] A. K. Biswas and S. K. Sarkar: "An arithmetic logic unit of a computer based on single electron transport system": Semiconductor Physics, Quantum Electronics & Opt-Electronics. 2003. Vol 6. No.1, pp 91-96

[7] A.K. Biswas and S. K. Sarkar: "Error Detection and Debugging on Information in Communication System Using Single Electron Circuit Based Binary Decision Diagram." Semiconductor Physics Quantum electronics and opt electronics, Vol. 6, pp.1-8, 2003

[8] Alexander N. Korotkov, "Single-electron logic and memory devices" INT. ELECTRONICS, 1999, Vol. 86, No. 5, 511- 547

[9] Casper Lageweg, Student Member, IEEE, Sorin Cot¸ofan˘a, Senior Member, IEEE, and Stamatis Vassiliadis, Fellow, IEEE "Single Electron Encoded Latches and Flip-Flops" IEEE TRANSACTIONS ON NANOTECHNOLOGY, VOL. 3, NO. 2, JUNE 2004

[10] C. Lageweg, S. Cot¸fan˘a, and S. Vassiliadis, "A linear threshold gate implementation in single electron technology," in IEEE Computer Society VLSI Workshop, Apr. 2001, pp. 93– A. Korotkov, "Single-electron logic and memory devices," Int. J. Electron., vol. 86, no. 5, pp. 511–547, 1999.

[11] K. Likharev, "Single-electron devices and their applications," Proc. IEEE, vol. 87, pp. 606–632, Apr. 1999.

[12] A. Korotkov, R. Chen, and K. Likharev, "Possible performance of capacitively coupled single-electron transistors in digital circuits," J. Appl. Phys., vol. 78, pp. 2520–2530, Aug. 1995.

[13] J. R. Tucker, "Complementary digital logic based on the "Coulomb blockade"," J. Appl. Phys., vol. 72, no. 9, pp. 4399–4413, Nov. 1992.

[14] A. Korotkov and K. Likharev, "Single-electron-parametron-based logic devices," J. Appl. Phys., vol. 84, no. 11, pp. 6114–6126, Dec. 1998.

[15] C. Wasshuber, H. Kosina, and S. Selberherr, "SIMON—A simulator for single-electron tunnel devices and circuits," IEEE Trans. Computer- Aided Design, vol. 16, pp. 937–944, Sept. 1997.

[16] A. B. Zorin, S. V. Lotkhov, H. Zangerle, and J. Niemeyer, "Coulomb blockade and cotunneling in single electron circuits with on-chip resistors: Toward the implementation of the R pump," J. Appl. Phys., vol. 88, no. 5, pp. 2665–2670, Sept. 2000.

[17] Z. Durrani, A. Irvine, and H. Ahmed, "Coulomb blockade memory using integrated single-electron transistor/metal–oxide–semiconductor transistor gain cells," IEEE Trans. Electron Devices, vol. 47, pp. 2334–2339, Dec. 2000.

[18] J. Millman and C. C. Halkias; Integrated Electronics- Analog and Digital Circuits and Systems_ McGraw Hill Education; 2 edition

[19] Millman's Electronic Devices & Ciruits 4th Edition (English, Paperback, Millman Jacob)

BIOGRAPHY

Anup Kumar Biswas is an Assistant Professor in the Department of Commuter Science and Engineering in Kalyani Govt. Engineering College. He is awarded his PhD [Engg.] degree in the stream of Electronics and Telecommunication Engineering from Jadavpur University in the year 2006. He has engaged in teaching and research activities since the last 16 years. His research interests include computer arithmetic, parallel architectures, nanotechnology, and computer-aided design. Dr. Biswas has published several papers in various national, international conferences and journals.